Lecture 15: Introduction to 3D Graphics and 3D Modeling
Logistics

- Project proposal deadline tomorrow midnight. Submit a one page write-up. Talk to me if you still haven’t!
- PA1 is due 10/30, 11:30 AM.
Where Are We?

Physical Scene
(Objects, lights)

Optics
(Gather light)

Image Sensing
(Optical to electrical signal transformation)

Image Signal Processing
(Signal reconstruction)

Computer Vision
(Semantics understanding)

Modeling
(Scene, optics)

Computer Graphics
(Visibility + Shading)

Display
(Generating lights)

Human Vision System
(Eye, visual cortex)

Video/Image De/Compression

Cloud/Storage
Why do those images look “real” to you?
Graphics Rendering

- Why do those images look “real” to you?
- Photorealism: a rendered image that looks like taken by a camera
Why do those images look “real” to you?
Photorealism: a rendered image that looks like taken by a camera
Why do those images look “real” to you?
Photorealism: a rendered image that looks like taken by a camera
Simulating the camera pipeline (sometimes assuming perfect optical to electrical signal transformation)
Could simulate some desired optical distortions
Graphics Rendering

Physical Scene
(Objects, lights)

Optics
(Gather light)

Image Sensing
(Optical to electrical signal transformation)

Image Signal Processing
(Signal reconstruction)

Display
(Generating lights)

Human Vision System
(Eye, visual cortex)

Simulate object behaviors, light-object interactions, light-lens interactions
Graphics Rendering

Physical Scene
(Objects, lights)

Optics
(Gather light)

Image Sensing
(Optical to electrical
signal transformation)

Image Signal Processing
(Signal reconstruction)

Computer Vision
(Semantics understanding)

Modeling
(Scene, optics)

Rendering
(Visibility + Shading)

Display
(Generating lights)

Human Vision System
(Eye, visual cortex)
Goal of Graphics and Rendering?

- Rendering 2D images from 3D scenes.
- Computer-generated imageries (CGI) are not always physically “real” (e.g., game/film), but should look photorealistic.

Graphics

Modeling → Lighting, Camera, and Material → Rendering

http://www.cgarena.com/freestuff/tutorials/max/thomas_highway/sergeant.html
Fundamental Trade-off

- Photo-realism vs. rendering speed
- How do we simulate light?
  - Atom/Molecule-level?
  - Radiometry/photometry?
  - Other high-level abstractions?
- How do we simulate lens?
  - Pinhole camera?
  - Thin lens?
  - Geometrical optics vs. wave optics?
3D Modeling

Parametric/Implicit surface

Polygon mesh

Point cloud

https://twitter.com/dmitriatsafe/status/1066037693174341632
https://commons.wikimedia.org/wiki/File:Sphère_-_monochrome_simple.svg
Mesh

https://www.geom.at/terrain-triangulation/
http://www.cgarena.com/freestuff/tutorials/max/thomas_highway/sergeant.html
Mesh

https://en.wikipedia.org/wiki/Polygon_mesh
Calculating the normal of a face by taking the **cross-product** of two vectors on the face.

[Link to Wikipedia](https://en.wikipedia.org/wiki/Normal_(geometry))
General Idea of Representing a Mesh

Encoding the coordinates of each vertex

\[ \text{V0} \{1, 1, 2\}, \text{V1} \{-1, 2, 1\}, \text{V2} \{0, 1, 0\}, \text{V3} \{-1, -1, 2\}, \text{V4} \{1, -1, 2\} \]

Encoding vertices of each face

\{V0, V2, V1\}, \{V1, V2, V4, V3\}

Could also have a normal vector list and various other primitives

RenderMan-like Interface

PointsPolygon [3, 4] [0, 2, 1, 1, 2, 4, 3] “P”

[1, 1, 2, -1, 2, 1, 0, 1, 0, -1, -1, 2, 1, -1, 2]
Polygon Triangulation

- Converting a polygon into a set of triangles.
  - Why? It usually allows more efficient implementations to have just one kind of primitive.

A “fan triangulation” that converts a convex N-gon into N-2 faces.

Ear-clipping triangulation.

https://en.wikipedia.org/wiki/Polygon_triangulation
Parametric Curve

- Parametric curve: defining a curve by evaluating a set of *parametric equations*. Each equation has 1 parameter. \( \mathbb{R} \rightarrow \mathbb{R}^2 \).

<table>
<thead>
<tr>
<th>Parabola ( y = x^2 )</th>
<th>Ellipse ( \frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x = t )</td>
<td>( x = a \cos(t) )</td>
</tr>
<tr>
<td>( y = t^2 )</td>
<td>( y = b \sin(t) )</td>
</tr>
</tbody>
</table>

To define a curve, the parametric equations have **1 parameter** \( t \).

http://amsi.org.au/ESA_Senior_Years/SeniorTopic2/2a/2a_2content_1.html
Parametric Surface

- Parametric surface: defining a surface by evaluating a set of \textit{parametric equations}. Each equation have 2 parameters. $\mathbb{R}^2 \rightarrow \mathbb{R}^3$.

Spherical surface with a fixed radius $\rho$

\begin{align*}
x &= \rho \sin \phi \cos \theta \\
y &= \rho \sin \phi \sin \theta \\
z &= \rho \cos \phi
\end{align*}

### Parametric Surface

- **Parametric surface**: defining a surface by evaluating a set of *parametric equations*. Each equation have 2 parameters. \( \mathbb{R}^2 \to \mathbb{R}^3 \).

#### Spherical surface with a fixed radius \( \rho \)

- \( x = \rho \sin \phi \cos \theta \)
- \( y = \rho \sin \phi \sin \theta \)
- \( z = \rho \cos \phi \)

#### Cylindrical surface with a fixed radius \( \rho \)

- \( x = \rho \cos \phi \)
- \( y = \rho \sin \phi \)
- \( z = z \)

Parametric Surface

- Parametric surface: defining a surface by evaluating a set of parametric equations. Each equation have 2 parameters. $\mathbb{R}^2 \rightarrow \mathbb{R}^3$.

**Spherical surface with a fixed radius $\rho$**

\[
\begin{align*}
x &= \rho \sin \phi \cos \theta \\
y &= \rho \sin \phi \sin \theta \\
z &= \rho \cos \phi
\end{align*}
\]

**Cylindrical surface with a fixed radius $\rho$**

\[
\begin{align*}
x &= \rho \cos \phi \\
y &= \rho \sin \phi \\
z &= z
\end{align*}
\]

To define a surface, the parametric equations have **2 parameters**.

Bezuer curve is a kind of parametric curve expressed as a weighted sum of a set of control points.

Weights are given by the Bernstein polynomial. Two end control points are on the curve.

Linear Bezier curve

\[ P = (1 - t) P_0 + t P_1 \]

https://en.wikipedia.org/wiki/B%C3%A9zier_curve
Bezier Curve

- Bezier curve is a kind of parametric curve expressed as a weighted sum of a set of control points.

  - Weights are given by the Bernstein polynomial. Two end control points are on the curve.

Linear Bezier curve

\[ P = (1 - t) P_0 + t P_1 \]

https://en.wikipedia.org/wiki/B%C3%A9zier_curve
Beziers Curve

Quadratic Bezier curve

- $P_A = (1 - t) P_0 + t P_1$
- $P_B = (1 - t) P_1 + t P_2$
- $P = (1 - t) P_A + t P_B$

Quadratic bezier curve has 3 control points. The curve changes with the control points.
Beziers Curve

Quadratic Bezier curve has 3 control points. The curve changes with the control points.

\[ P_A = (1 - t)P_0 + tP_1 \]
\[ P_B = (1 - t)P_1 + tP_2 \]
\[ P = (1 - t)^2P_0 + 2t(1-t)P_1 + t^2P_2 \]
A bezier curve of order n has \((n+1)\) control points.

\[
P = (1 - t)^3 P_0 + 3(1 - t)^2 t P_1 + 3(1 - t)t^2 P_2 + t^3 P_3
\]

https://en.wikipedia.org/wiki/B%C3%A9zier_curve
A bezier curve of order n has (n+1) control points.

\[ P = (1 - t)^3 P_0 + 3(1 - t)^2 t P_1 + 3(1 - t)t^2 P_2 + t^3 P_3 \]
A Bezier curve of order \( n \) has \( (n+1) \) control points.

Beziers Curve

- Third-order Bezier curve
- Fourth-order Bezier curve

https://en.wikipedia.org/wiki/B%C3%A9zier_curve
A bezier curve of order n has (n+1) control points.

Third-order Bezier curve

Fourth-order Bezier curve

https://en.wikipedia.org/wiki/B%C3%A9zier_curve
Beziers Curve

$$B(t) = \sum_{i=0}^{n} b_{i,n}(t) P_i$$

$$b_{i,n}(t) = \binom{n}{t} t^i (1-t)^{n-i}$$

Basis functions: Bernstein polynomials

Weights

Control points

Parameter

Binomial coefficients

The first and last control points are also the end points of the generate curve.

https://en.wikipedia.org/wiki/B%C3%A9zier_curve
A (m, n) Bezier surface defines a surface where every point is a weighted sum of (m+1) x (n+1) control points.

\[ p(u, v) = B^3_0(u)B^3_0(v)P_{00} + B^3_1(u)B^3_0(v)P_{01} + B^3_2(u)B^3_0(v)P_{02} + B^3_3(u)B^3_0(v)P_{03} \\
+ B^3_0(u)B^3_1(v)P_{01} + B^3_1(u)B^3_1(v)P_{11} + B^3_2(u)B^3_1(v)P_{12} + B^3_3(u)B^3_1(v)P_{13} \\
+ B^3_0(u)B^3_2(v)P_{02} + B^3_1(u)B^3_2(v)P_{12} + B^3_2(u)B^3_2(v)P_{22} + B^3_3(u)B^3_2(v)P_{23} \\
+ B^3_0(u)B^3_3(v)P_{03} + B^3_1(u)B^3_3(v)P_{13} + B^3_2(u)B^3_3(v)P_{23} + B^3_3(u)B^3_3(v)P_{33} \]
A (m, n) Bezier surface defines a surface where every point is a weighted sum of (m+1) x (n+1) control points.

\[ p(u, v) = B_0^3(u)B_0^3(v)P_{00} + B_1^3(u)B_0^3(v)P_{01} + B_2^3(u)B_0^3(v)P_{20} + B_3^3(u)B_0^3(v)P_{30} \]
\[ + B_0^3(u)B_1^3(v)P_{01} + B_1^3(u)B_1^3(v)P_{11} + B_2^3(u)B_1^3(v)P_{21} + B_3^3(u)B_1^3(v)P_{31} \]
\[ + B_0^3(u)B_2^3(v)P_{02} + B_1^3(u)B_2^3(v)P_{12} + B_2^3(u)B_2^3(v)P_{22} + B_3^3(u)B_2^3(v)P_{32} \]
\[ + B_0^3(u)B_3^3(v)P_{03} + B_1^3(u)B_3^3(v)P_{13} + B_2^3(u)B_3^3(v)P_{23} + B_3^3(u)B_3^3(v)P_{33} \]

\[ p(u, v) = [(1 - u)^3, 3u(1 - u)^2, 3u^2(1 - u), u^3] \times \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} \times \begin{bmatrix} (1 - v)^3 \\ 3v(1 - v)^2 \\ 3v^2(1 - v) \\ v^3 \end{bmatrix} \]
A (m, n) Bezier surface defines a surface where every point is a weighted sum of (m+1) x (n+1) control points.

A (3, 3) Bezier surface/patch

https://en.wikipedia.org/wiki/B%C3%A9zier_surface
A \((m, n)\) bezier surface defines a surface where every point is a weighted sum of \((m+1)\times(n+1)\) control points.

\[
p(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_i^n(u)B_j^m(v)
\]

\[
B_i^n(u) = \binom{n}{i} u^i (1-u)^{n-i}
\]

\[
\binom{n}{i} = \frac{n!}{i!(n-i)!}
\]
A Bezier surface could be used directly for rendering, but it is usually converted to a mesh (a form of tessellation), which is easier to render.

Sample the Bezier surface by calculating a set of points, which become mesh vertices. Connect these vertices generates a mesh. Sampling could be uniform, or could be non-uniform (e.g., more sample when curvature is higher).

https://en.wikipedia.org/wiki/B%C3%A9zier_surface
Point Cloud/Point Set

Laser Scanner

From Mike Jarvis, Prof. @ History Dept. Scanning Elmina Castle, a UNESCO World Heritage Site at Ghana.

LiDAR

Velodyne LiDAR (Light Detection and Ranging)

Uber's self-driving car equipped with LiDARs

https://news.voyage.auto/an-introduction-to-lidar-the-key-self-driving-car-sensor-a7e405590cff7qi-78d02169e6e

https://www.iftas.de/Main/VelodyneHDL64E?setlang=en
LiDAR

Tilting mirror
Optical rotary encoder
Servo motor
Laser source
Objects
Optical rotary encoder
Receiver
iPhone 12 Pro LiDAR sensor allows for 6x faster low-light autofocus, instant AR
iPhone 12 Pro LiDAR sensor allows for 6x faster low-light autofocus, instant AR

https://appleinsider.com/articles/20/10/13/iphone-12-pro-lidar-sensor-allows-sixx-faster-low-light-autofocus-instant-ar


What are these cameras?
Point Set Triangulation

- Point set could be used directly for rendering (point-based graphics), but it is usually converted to a mesh, which is easier to render.

Delaunay triangulation:
no point is inside the circumcircle of any triangle.

https://www.semanticscholar.org/paper/Triangulating-Point-Clouds-with-Spherical-Topology-Hormann-Reimers/94d49052e51e5ca3e7b18b7897b51826e359db
Reference Material

- Tessellation of 4x4 Bezier Patches for the Intel Pentium III Processor
- Bézier Curves and Surfaces: the Utah Teapot. A series of articles from scratchapixel on Bezier curves and surfaces.